Git/ GitHub Guide

🥋 Git & GitHub Master Notes

Video: 🎥 [YouTube: Git Basics Explained]

🧭 0. General Flow

Typical safe workflow for any project:

1. `git status` → check what's happening
2. `git add .` → stage your changes
3. `git commit -m "meaningful message"`
4. `git fetch origin` → check for remote updates
5. `git pull origin main --rebase` → safely pull latest changes
6. `git push origin main` → push your code

💡 *Never force-push (`-f`) unless you are 100% sure you want to overwrite remote history.*

📁 1. Repository Setup

Command	Description
`git init`	Initialize Git in current folder (makes it a “working directory”)
`git status`	Shows current tracked/untracked files
`git add <file>` / `git add .`	Stage changes
`git rm --cached <file>`	Unstage a file
`git restore <file>`	Discard local uncommitted changes
`git commit -m "msg"`	Save changes to local repo
`git log` / `git log --oneline`	Show commit history

🗒️ **Extra:**

- `.gitignore` → files/folders Git should not track
- `.gitkeep` → empty file to make Git track an otherwise empty folder

---

🌿 2. Branching & Merging

Command	Description
`git branch`	List branches
`git branch new-branch`	Create new branch
`git switch new-branch`	Switch to branch
`git switch -c new-branch`	Create & switch
`git checkout branch`	Switch (older syntax)
`git merge branch-name`	Merge branch into current
`git merge --abort`	Cancel a merge conflict
`git branch -m old new`	Rename branch
`git branch -d branch`	Delete branch

🧩 3. Inspecting & Comparing

Command	Description
`git diff`	Show unstaged differences
`git diff --staged`	Compare staged vs last commit
`git diff branch1 branch2`	Compare two branches
`git reflog`	Show full action history (life-saver!)
`git reset --hard <commit>`	Revert repo to a specific commit *(⚠ irreversible)*

💼 4. Stashing

Temporarily save work without committing:

Command	Description
`git stash`	Stash changes
`git stash save "msg"`	Stash with note
`git stash list`	Show stashes
`git stash apply` / `git stash pop`	Re-apply stashed work
`git stash drop`	Delete one stash
`git stash clear`	Delete all stashes

🔁 5. Rebasing (⚠ Advanced)

Rebase = move/redo commits from one branch onto another base commit.

Command	Description
`git rebase branch-name`	Apply commits on top of given branch
`git rebase --abort`	Stop an in-progress rebase

⚠ Do not rebase on `main` unless you fully understand what you're doing.

☁️ 6. Remote Setup & Push

Command	Description
`git remote add origin <repoURL>`	Connect local repo to GitHub
`git remote -v`	Verify connection
`git push -u origin main`	First push
`git push`	Later pushes
`git pull --rebase origin main`	Pull safely (avoids merge mess)
`git remote set-url origin <newURL>`	Fix remote if changed

Common issue: "remote: Repository not found" → verify URL & access rights
📖 [GFG Article]

🧱 7. Safe Push Procedure (Never lose your code again)

1. Always check: `git status`
2. Stage → `git add .`
3. Commit → `git commit -m "msg"`
4. Pull safely → `git pull --rebase origin main`
5. Push → `git push origin main`
6. Never force-push unless you *intend* to replace remote history.

🛡️ 8. Protecting Your Main Branch (GitHub)

1. On GitHub → **Settings** → **Branches** → **Add Rule**
2. Branch pattern → `main`
3. Enable:
   - ✅ Require a pull request before merging
   - ✅ Require status checks to pass
   - 🚫 Do not allow force pushes
   - 🚫 Do not allow deletions
4. Save rule.

Now, even you can't accidentally nuke `main`.

💾 9. Auto-Backup Setup (GitHub Actions)

Create a private backup repo (e.g., `mygame-backup`)
Then in your main repo:

Add secret:

- Go to → Settings → Secrets → Actions
- Add new secret → `BACKUP_TOKEN` → paste your Personal Access Token (with `repo` + `workflow` scopes)

Create workflow:
File → `.github/workflows/auto-backup.yml`

```
name: Auto Backup to Secondary Repo
on:
  push:
    branches:
      - main
      - "**"

jobs:
  backup:
    runs-on: ubuntu-latest
    steps:
      - name: Checkout repo
        uses: actions/checkout@v4
        with:
          fetch-depth: 0
      - name: Mirror to backup repo
        run: |
          git remote add backup https://x-access-token:${{ secrets.BACKUP_TOKEN
}}@github.com/<your-username>/mygame-backup.git
          git push --mirror backup
```

✅ Every push = automatic mirror backup to your secondary repo.

🧠 10. Optional: Local Backups

Manual weekly backup:

```
git bundle create backup-$(date +%Y%m%d).bundle --all
```

Restore later:

```
git clone backup-YYYYMMDD.bundle project-folder
```

🪄 Bonus Pro Tips

- `git log --graph --oneline --decorate` → beautiful branch tree view
- `git restore .` → discard all local changes
- `git tag v1.0` → mark releases
- `git push origin --tags` → push all tags